

**CIRCULATION COPY**  
**SUBJECT TO RECALL**  
**IN TWO WEEKS**

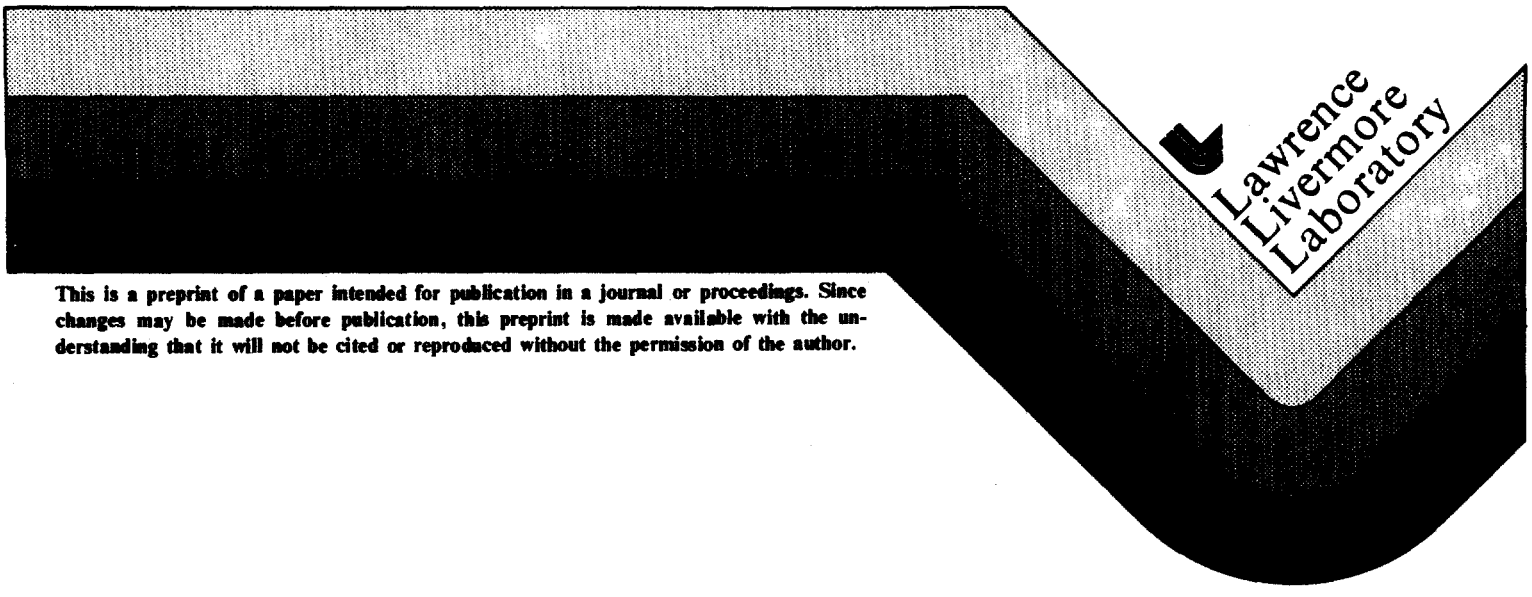
**UCRL-83349**  
**PREPRINT**

**2-D DIGITAL SIGNAL PROCESSING WITH AN ARRAY PROCESSOR**

**Richard E. Twogood**

This paper was prepared for submittal to  
1980 IEEE International Conference on  
Acoustics, Speech, and Signal Processing  
Denver, Colorado, April 1980

January 1980

The logo for Lawrence Livermore Laboratory is a large, stylized 'V' shape. The top horizontal bar of the 'V' is light gray, while the two slanted sides are dark gray. The text 'Lawrence Livermore Laboratory' is written in a sans-serif font, following the curve of the right slanted side. A small, solid black arrow points downwards from the top horizontal bar towards the text.

**Lawrence  
Livermore  
Laboratory**

This is a preprint of a paper intended for publication in a journal or proceedings. Since changes may be made before publication, this preprint is made available with the understanding that it will not be cited or reproduced without the permission of the author.

#### DISCLAIMER

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

## 2-D DIGITAL SIGNAL PROCESSING WITH AN ARRAY PROCESSOR

Richard E. Twogood

Lawrence Livermore Laboratory, University of California  
Livermore, California

### ABSTRACT

The applicability of array processor (AP) technology to 2-D digital signal processing is investigated in this paper. The AP-based image processing facility at LLL is described, with emphasis on our applications software package which utilizes the array processor. Implementations of several key image processing algorithms are discussed and compared with other conventional processors, indicating that array processors are extremely cost-effective for image processing applications.

development of alternative energy sources and weapons research, although a large number of other programs also exist. In such an applied physics setting, a number of applications have arisen requiring the processing of two-dimensional imagery, or image processing [1]. Some of the image processing applications we have recently investigated include enhancement and restoration procedures for flash x-ray radiographs, enhancement of pinhole camera diagnostic images from laser fusion experiments and nuclear weapons tests, aerial surveillance for test ban treaty verification, decoding of Fresnel zone-plate coded images for laser diagnostics, ultrasonic imaging for biological applications, and many others.

### INTRODUCTION

One of the most important developments in the areas of high-speed scientific computing and distributed processing has been the recent introduction of the array processor. An AP is an arithmetic processor which, when connected to a host computer, provides extremely fast floating point computation. This is achieved via the use of parallel, pipelined floating point units and very fast registers, data memory, and program memory.

Digital image processing is an example of one application area particularly well-suited to array processor technology. This is due not only to the tremendous amount of numerical computation required, but also because most of the practical image processing algorithms are well-suited to implementation on a vector or parallel processor. In this paper we describe our efforts in developing a general purpose image processing facility which utilizes these capabilities of the array processor.

### ARRAY PROCESSOR BASED IMAGE PROCESSING FACILITY

The Lawrence Livermore Laboratory is a national laboratory funded by the Department of Energy. The major emphasis at LLL is on the

To provide effective image processing capabilities for such a diverse set of applications, we have developed an interactive minicomputer/array processor-based image processing facility. A block diagram of our system is shown in Figure 1. Virtually all of the numerical computations required by the image processing algorithms are performed on the array processor, which is a Floating Point Systems AP120B with 16k of 167ns data memory (soon to be upgraded to 64k). The minicomputer, a PDP 11/70, serves primarily as a controller to drive the image display system, initiate array processor activity, and communicate with LLL's large computer network ("Octopus") consisting of several large mainframe computers.

One unique characteristic of our system as shown in Figure 1 is the existence of a direct data path between the array processor and the 80 megabyte system disks. This was achieved by modifying both the array processor executive software (supplied by Floating Point Systems) and the PDP 11/70's operating system's disk driver software to allow for this "device to device" transfer. Such a configuration provides a very significant savings in input/output (I/O) requirements when compared with the more standard configuration of Figure 2a. This savings results for two reasons: 1) the larger amount of AP data memory (versus 11/70 available buffer) reduces the number of disk accesses required, and 2) the dual transfer of data (disk to host, then host to AP) required by the configuration of Figure 2a has been simplified to just one data transfer (disk to AP). Of course, an alternate method of achieving this savings is to purchase dedicated disks for the

Work performed under the auspices of the U. S. Department of Energy by the Lawrence Livermore Laboratory under contract number W-7405-ENG-48 and supported in part by NSF Grant ENG-78-04240.

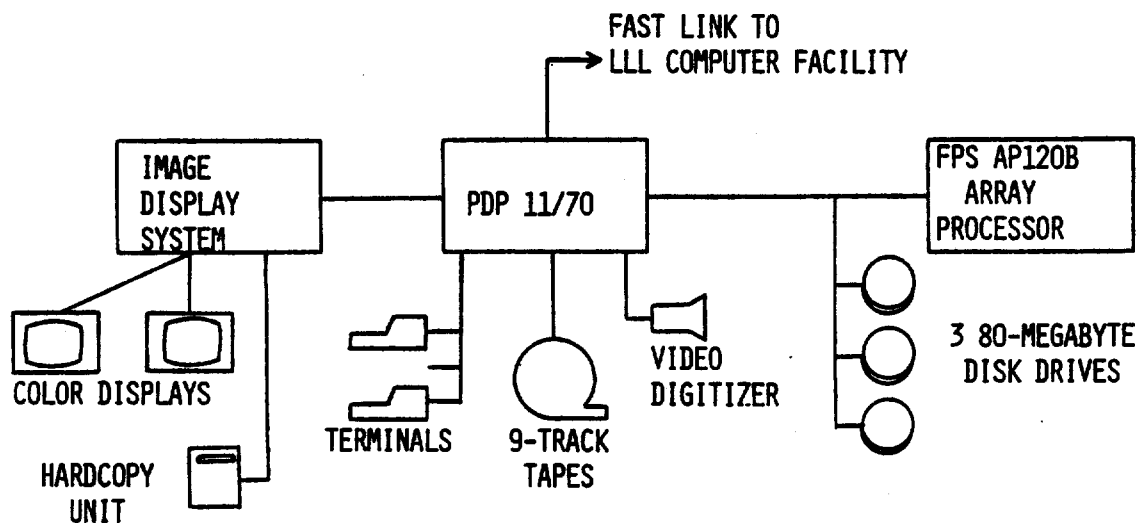


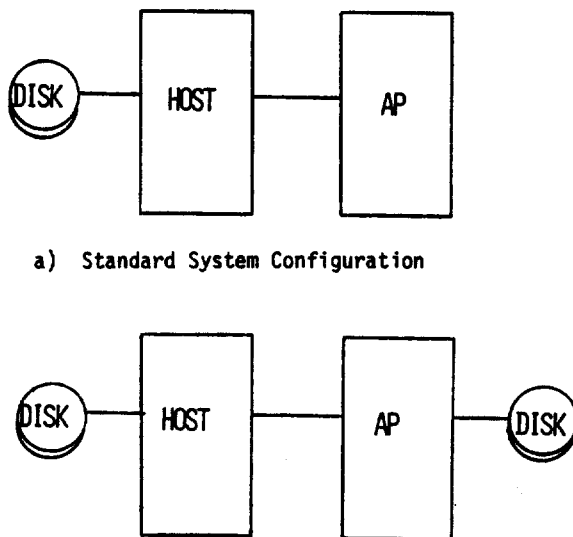
Figure 1. LLL's Image Processing Research Facility

## 2-D SIGNAL PROCESSING APPLICATIONS SOFTWARE

As indicated above, we are involved in a wide variety of applications at LLL requiring 2-D digital signal processing capabilities. Due to the diversity of the application needs, we have developed a general-purpose software system called "IMG" which implements a large number of fundamental image processing algorithms on our minicomputer and array processor. While there are too many routines to mention individually, most of the capabilities of IMG can be categorized as follows:

- 1) Pointwise operators on images (includes add, subtract, multiply, divide images; clip, exponentiate, log, absolute value, scale, and compute statistics on an image; histogram)
- 2) "Moving window" operators (includes median window filter, Wallis' contrast stretch algorithm [2], Lee's contrast stretch algorithm [3], nonrecursive filtering via convolution, derivatives, and average filter)
- 3) Global operators (includes 2-D FFT, 2-D convolution via the FFT, alpha root, histogram equalization)
- 4) Control functions (includes file name manipulation, file documentation via trailers, format conversions, image subsampling, image synthesis, and many others).

In addition, we are continuing to develop additional image processing software routines, many of which are based on the above fundamental operators. Examples include some linear and nonlinear filtering techniques and several image restoration algorithms, including Wiener restoration, spectral equalization, Frieden's iterative technique, and maximum entropy restoration. All of these software routines operate on disk-based images which, with a few exceptions such as radix-2 2-D FFT applications, are of arbitrary dimensions.



a) Standard System Configuration

b) System with Dedicated Disk(s) for AP

Figure 2. Typical Array Processor System Configurations

AP, as shown in Figure 2b. By implementing our "device to device" transfer capability, however, we have avoided the increased capital investment (in the form of additional disks and an I/O processor for the AP) such a system would require.

### SOME IMPLEMENTATION AND AP PERFORMANCE ISSUES

A detailed discussion of the implementation issues encountered (e.g. CPU vs I/O effects, data memory size, algorithm vectorizability, etc.) in 2-D signal processing with an AP is well beyond the scope of the present paper. However, we will attempt to demonstrate the performance capabilities of the AP image processing package by briefly discussing a few key algorithms below.

**2-D FFT of Disk-based Images:** The 2-D FFT is, of course, one of the most important algorithms used in image processing. It is utilized in performing nonrecursive linear filtering, in several nonlinear filtering techniques, and in image restoration, to mention a few. The historical approach for computing 2-D FFT's of arrays larger than primary storage consists of decomposing the 2-D FFT into iterated 1-D FFT's. The calculation proceeds as follows: first each row in the array is transformed with the 1-D FFT. The array is then transposed and the row transform process is repeated (this computes the column transforms of the original array). This three-step procedure yields the transpose of the transformed array.

A significant part of the data manipulation required by the above procedure is a result of the matrix transpositions. A fast matrix transposition scheme, originally proposed by Eklundh in [4] and further extended in [5], provides an efficient (particularly in terms of I/O) algorithm for transposing matrices of dimension  $2^n \times 2^n$ . An analysis of CPU and I/O requirements for 2-D digital filtering via the 2-D FFT with an array processor is given in [6], where the extended Eklundh method of matrix transposition was applied to the array processor system configuration of Figure 2a.

Two alternative schemes for computing the 2-D FFT without a matrix transposition have been developed in [7] and [8]. Careful examination of those algorithms, however, shows that they require exactly the same I/O procedure as Eklundh's matrix transposition scheme. This is a result of the similarity between data requirements for the matrix transposition and the column FFT. In fact, the 2-D FFT without matrix transposing may require slightly more I/O if the bit-reversed ordering of the rows that naturally results in the spatial frequency domain cannot be tolerated.

Until recently, the I/O scheme used in [4]-[5] and [7]-[8] for the implementation of 2-D FFTs (with or without matrix transposing) was the best available technique. A recently proposed "two-level" algorithm for matrix transposition, however, offers a significantly improved I/O scheme. This two-level technique is essentially Eklundh's method with a re-ordering of the sequence of computation, with the re-ordering resulting in fewer single-line transputs and hence fewer disk accesses. A detailed description of our array processor

	2-D FFT (512 x 512 real)	Contrast Stretch (512 x 512 Image, 2 x 2 Window)
PDP 11/70	122 sec (83, 39)	~3 minutes
PDP 11/70, AP120B with 16K mem.	18 sec (4.5, 13.5)	11.0 sec (7.0, 4.0)
PDP 11/70, AP120B with 64K mem.	~8 sec	~9.3 sec (7.0, 2.3)
CDC 7600 with 512K	3.3 sec (2.1, 1.2)	5.1 sec (3.9, 1.2)

TABLE 1. Comparative timings for 2 representative image processing algorithms on various computer systems. Approximate timings (~) denoted where hardware or software was insufficient for accurate timing. Below each time is given the (CPU, I/O) portion of the total.

implementation of this scheme for the 2-D FFT is given in [9]. As noted in Table 1, our 512 x 512 FFT (with 16k AP data memory) requires only 18 seconds, providing highly interactive image processing. Upgrading to 64k of AP data memory will significantly decrease the FFT time, to about 8 seconds. The array processor implementation therefore far exceeds the minicomputer-alone implementation, and even approaches that of the CDC 7600. This is remarkable when one considers that the 7600, at about \$8 million, costs approximately 100 times as much as the array processor!

**Contrast stretch algorithm:** As noted above, we have implemented several "moving window" operators on our array processor. These image processing algorithms use the pixels in a 2-D neighborhood of the current input pixel to calculate the current output pixel. Examples of such algorithms are FIR filters and the median window filter. Typical window sizes range from 2 x 2 to 25 x 25 or larger, with the smaller windows being of more interest for most practical image processing problems.

Wallis' contrast stretching algorithm [2] is one particularly effective moving-window enhancement technique. The algorithm is a nonlinear, image-dependent processor which equalizes the local mean and local variance throughout an image. Our array processor implementation of this algorithm, as shown in Table 1, requires only 11 seconds for a 2 x 2 window and a 512 x 512 image. Once again, the

array processor provides an interactive capability that approaches the performance of the far more expensive mainframe computer.

#### CONCLUSION

The LLL investigation into the application of array processors for interactive image processing has been summarized. Our image processing facility has been described, with emphasis on the role of the array processor. A brief discussion of algorithm implementations was given, with examples illustrating that the array processor can provide the computational throughput required for highly interactive image processing. For the typical image processing algorithms discussed, the array processor was found to deliver approximately 50% of the capability of a CDC 7600 at about 1% of the cost.

#### REFERENCES

- [1] R. E. Twogood, "Digital Image Processing at LLL," Energy and Technology Review, April 1979.
- [2] R. Wallis, "A Fast Algorithm for Space Variant Contrast Stretching," Proc. of the 11th Asilomar Conf. on Circ., Systems, and Computers, Nov., 1977, pp. 240-245.
- [3] J. S. Lee, "Digital Image Processing by Use of Local Statistics," Proc. IEEE Comp. Soc. Conf. on Pattern Recog. and Image Processing, Chicago, Ill., May 1978, pp. 55-61.
- [4] J. O. Eklundh, "A Fast Computer Method for Matrix Transposing," IEEE Trans. Comput., Vol C-21, July 1972, pp 801-803.
- [5] R. E. Twogood and M. P. Ekstrom, "An Extension of Eklundh's Matrix Transposition Algorithm and Its Application in Digital Image Processing," IEEE Trans. Comp., Vol. C-25, September 1976, pp. 950-952.
- [6] R. E. Twogood, "Array Processor Implementation of 2-D Digital Filters," Proc. of the 1979 IEEE International Conference on Circuits and Systems, Tokyo, Japan, July 17-19, 1979.
- [7] M. Onoe, "A Method for Computing Large-Scale Two-Dimensional Transform Without Transposing Data Matrix," Proc. IEEE, Vol. 63, No. 1, January 1975, pp. 196-197.
- [8] D. B. Harris, et al., "Vector Radix Fast Fourier Transform," Proc. 1977 IEEE Inter. Conf. on Acoustics, Speech, and Signal Processing, Hartford, Connecticut, May 1977, pp. 548-551.

- [9] R. E. Twogood, "2-D FFTs of Large Images with the AP1208," 1980 Floating Point Systems User's Group Meeting, San Francisco, CA, May 1980.

#### NOTICE

"This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Department of Energy, nor any of their employees, nor any of their contractors, subcontractors, or their employees, make any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately-owned rights."

Reference to a company or product name does not imply approval or recommendation of the product by the University of California or the U.S. Department of Energy to the exclusion of others that may be suitable.